

## CELLULAR TELEPHONY SEARCHER

## FIELD AND BACKGROUND OF THE INVENTION

The present invention relates to cellular telephony and, more particularly, to a searcher for a DSSS cellular telephony system.

In a DSSS cellular telephony system, the base stations identify themselves by transmitting pilot signals. Each pilot signal is a sequence of zero bits, modulated, according to the principles of DSSS encoding, by a pseudonoise (PN) sequence, or an extended pseudonoise sequence.

For example, under the IS-95 interim standard, the PN sequence is  $2^{15}$  chips long, with the  $n$ -th chip including an in-phase component  $i(n)$  and a quadrature component  $q(n)$ . The initial values of  $i$  and  $q$  are  $i(1)=q(1)=1$  and  $i(n)=q(n)=0$  for  $2 \leq n \leq 15$ . Subsequent values of  $i$  and  $q$ , up to  $n=2^{15}-1$ , are obtained recursively as follows:

$$i(n)=i(n-15)+i(n-10)+i(n-8)+i(n-7)+i(n-6)+i(n-2) \quad (1)$$

$$q(n)=q(n-15)+q(n-12)+q(n-11)+q(n-10)+q(n-9)+q(n-5)+q(n-4)+q(n-3) \quad (2)$$

where the additions are modulo 2. Finally,  $i(2^{15})=q(2^{15})=0$ .

The same PN sequence is used by each of the base stations. The base stations are synchronized; and each base station uses the PN sequence with a different delay (also called "PN offset") to produce the pilot signal. This enables the mobile units of the cellular telephony network to distinguish one base station from another.

The total signal received by a mobile station, as a function of time  $t$ , is:

$$RX(t) = \sum_{b=1}^B \sum_{m=1}^{M_b} C(b, m, t) \cdot PN(t + \text{offset}(b) + \tau(b, m)) \cdot \left[ 1 + \sum_{i=1}^h \alpha_i \cdot D(i, b, t) \cdot W(i, b) \right] + N \quad (3)$$

Here,  $b$  indexes the  $B$  base stations;  $m$  indexes the  $M_b$  transmission paths (multipath channels) from base station  $b$  to the mobile station;  $C$  is the channel gain of multipath channel  $m$ ;  $\tau$  is the additional delay introduced to the PN sequence by multipath channel  $m$ ; the "1" inside the brackets represents the sequence of zeros that is modulated by the base stations to produce the pilot signals;  $i$  indexes the  $I_b$  other users that are transmitting via base station  $b$  at time  $t$ ;  $\alpha_i$  is the power of user  $i$  relative to the pilot signal;  $D$  is the data transmitted by user  $i$ ;  $W$  is a code sequence (for example, a Hadamard code sequence) that is used in addition to the PN sequence to modulate data  $D$  and allow simultaneous transmission on the same physical channel by all the users in addition to the pilot signals; and  $N$  is additive noise.

Each mobile unit of the cellular telephony network determines which base station to communicate with (typically, the nearest base station) by correlating this signal with the PN sequence at a set of trial delays. Because data  $D$  are modulated by sequences  $W$ , the correlation of the part of the signal that comes from other users is negligible. The correlation with the pilot signals also is negligible, except at trial delays that are equal to the PN offsets used by the base stations, as modified by multipath delays  $\tau$ . Specifically, a pilot signal that arrives at a delay, that is equal to the sum of a base station offset and one of the multipath delays  $\tau$  associated with transmissions from that base station, gives a

significant contribution to the correlation at a matching trial delay; and all other pilot signals contribute negligibly to the correlation at that trial delay. This correlating is performed when the mobile station powers up, and continuously thereafter, to allow hand over from one base station to another when the mobile station crosses a cell boundary. The delays of the various base stations are well separated, by more than the largest anticipated multipath delay, so in the absence of additive noise and in the absence of multipath delays, only a small number of correlations, equal to the number of potential nearest base stations, would have to be performed, to identify the base station whose delay gives the highest correlation as the nearest base station. According to the IS-95 standard, this separation is at least 256 chip durations  $T_c$ . Because the pilot signals and data  $D$  are received by the mobile station from each base station via several paths at different delays (PN offset +  $\tau$ ), the various replicas of the signals thus received are combined to suppress the deterministic noise represented by the various multipath delays  $\tau$ . For example, maximal ratio combining is the optimal combination method in a bit error rate and frame error rate sense. In order to do this combining, the multipath delays must be determined. Therefore, the correlation is performed at a series of delays in a window centered on the nominal delay. The size of this window depends on the local topography, and is provided to the mobile unit by the base station. One typical window size, according to the IS-95 standard, is 60 chip durations.

FIG. 3 is a schematic block diagram of a mobile station receiver 30. RF signals are received by an antenna 60, down converted to an intermediate frequency (IF) by a down converter 62, filtered by a bandpass filter 64 (typically a surface acoustic wave filter) to eliminate signals outside the required bandwidth, and amplified by an automatic gain control 66. The amplified IF signals are multiplied by an IF sinusoid 65, without (block 68i) and with (block 68q) a 90° phase shift 67, to produce an in-phase signal  $I$  and a quadrature signal  $Q$ . In-phase signal  $I$  is filtered by a low-pass filter 70i and digitized by an A/D converter 72i. Similarly, quadrature signal  $Q$  is filtered by a low-pass filter 70q and digitized by an A/D converter 72q. A searcher 80 receives the digitized signals and performs the correlations needed to determine the various multipath delays  $\tau$  inside the target window. The digitized signals are again correlated, at the delays determined by searcher 80, by the correlators of a correlator bank 74, and the outputs of correlator bank 74 are combined, in a maximal ratio sense, in a rake combiner 76 to produce the final output signal.

In order to ensure uninterrupted communication as a mobile station crosses from one cell to another, the correlations performed by searcher 80 must be performed rapidly. In fact, it is not necessary to perform the full correlation at each delay in the window. It suffices to perform a correlation that is only long enough to ensure a high detection probability at the right delay and a low false alarm probability at the wrong delay. Typically, the length of the correlation, measured as a multiple  $N$  of the chip duration  $T_c$ , is between  $500T_c$  and  $2000T_c$ .

To make the correlations even more efficient, the dual dwell algorithm is used. At each delay in the window, the correlation is performed for a number  $M$  of chip durations that is less than  $N$ . Only if the correlation value after  $M$  chip durations exceeds a certain threshold is the correlation performed for the full  $N$  chip durations. The threshold, and the parameters  $N$  and  $M$ , are chosen to maximize the detection probability while minimizing both the false alarm probability and the time spent correlating. See, for example, M. K. Simon, J. K. Omura, R. A. Scholtz and B. K.

Levitt, *Spread Spectrum Communication, Vol. III*, Computer Science Press, 1989, chapter 1, particularly section 1.3, and D. M. Dicarlo and C. L. Weber, "Multiple dwell serial search: performance and application to direct sequence code acquisition", *IEEE Transactions on Communications* vol. COM-31 no. 5 pp. 650-659, May 1983. In the prior art implementation of this algorithm, several correlators are used by searcher 80 to correlate the received pilot signal with the PN sequence at several adjacent delays in the window. If none of the correlation values exceeds the threshold after M chip durations, then the correlators are used to correlate the received pilot signal with the PN sequence at the next several adjacent delays. If at least one of the correlation values exceeds the threshold after M chip durations, then all the correlations are continued for the full N chip durations, but only the correlation values obtained by the correlators whose correlation values exceeded the threshold after the initial M chip durations are actually considered. The brute force approach to reducing search time, adding more correlators, is inefficient, because the more correlators that are used, the more likely it is that one of the correlators passes the threshold. In that case, the other correlators, which did not pass the threshold, continue to correlate unnecessarily for the full N chip durations.

There is thus a widely recognized need for, and it would be highly advantageous to have, a configuration for a cellular telephony searcher that would allow the efficient use of many correlators.

#### BRIEF DESCRIPTION OF THE DRAWINGS

The invention is herein described, by way of example only, with reference to the accompanying drawings, wherein:

FIG. 1 is a partial block diagram of a searcher of the present invention,

FIG. 2 is a flow chart for the decision of whether to move a correlator to a new delay;

FIG. 3 (prior art) is a schematic block diagram of the receiver of a cellular telephony mobile unit.

#### DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention is of a cellular telephony searcher which can be used by a mobile station to identify the several strongest multipath components of nearby base stations faster than presently known searchers.

The principles and operation of a cellular telephony searcher according to the present invention may be better understood with reference to the drawings and the accompanying description.

Referring now to the drawings, FIG. 1 is a partial block diagram of a searcher 10 of the present invention. Searcher 10 includes a PN sequence generator 12, a delay line 14 that in turn includes several complex delay units 16, a multiplexer 18, several correlators 20, a hold unit 26 and a next location unit 28. With each correlator 20 is associated an index register 22 and a memory 24. Memory 24 includes several complex registers and several corresponding integer registers, as discussed below. For illustrational simplicity, only two correlators 20 are shown, and only six delay units 16 are shown in delay line 14. In practice, the preferred number of correlators 20 is at least 8. The preferred number of delay units 16 is discussed below.

Also shown in FIG. 1 is a receiver 30 and a clock 32.

Block 30 of FIG. 1 represents prior art receiver 30 of FIG. 3, except for searcher 80; and, in fact, according to the

present invention, searcher 10 substitutes directly for searcher 80 in receiver 30 of FIG. 3. The calculation performed by each correlator 20 is

$$S_k(v, \gamma) = \sum_{k=1}^K RX_{k+v} PN_k^{*} \quad (4)$$

where the  $RX_k$  are successive values of the received signal of equation (3), the  $PN_k$  are successive values of the PN sequence received by correlator 20 from PN sequence generator 12, and the summation index  $k$  runs from 1 to an upper limit  $K$ . The received signal is not necessarily sampled at the same rate as the PN sequence. In the examples presented herein, new samples  $RX_k$  are provided to correlators 20 by A/D converters 72 at time intervals of  $T_c/2$ . The parameter  $v$  represents the time at which the correlation performed by a particular correlator 20 starts. The parameter  $\gamma$  represents the delay at which the correlation is performed, relative to the time at which the correlation starts. The samples  $RX_k$  and  $PN_k$  are complex, and the asterisk represents complex conjugation:  $PN_k^*$  is the complex conjugate of  $PN_k$ . For example, in a searcher 10 with four correlators, the correlation performed initially by the first correlator 20 is:

$$S = RX(0)PN(0) + RX(T_c)PN(T_c) + RX(2T_c)PN(2T_c) + RX(3T_c)PN(3T_c) + \dots \quad (5)$$

the correlation performed initially by the second correlator 20 is:

$$S = RX(T_c/2)PN(0) + RX(3T_c/2)PN(T_c) + RX(5T_c/2)PN(2T_c) + RX(7T_c/2)PN(3T_c) + \dots \quad (6)$$

the correlation performed initially by the third correlator 20 is:

$$S = RX(T_c)PN(0) + RX(2T_c)PN(T_c) + RX(3T_c)PN(2T_c) + RX(4T_c)PN(3T_c) + \dots \quad (7)$$

and the correlation performed initially by the fourth correlator 20 is:

$$S = RX(3T_c/2)PN(0) + RX(5T_c/2)PN(T_c) + RX(7T_c/2)PN(2T_c) + RX(9T_c/2)PN(3T_c) + \dots \quad (8)$$

(In equations (5)-(8),  $RX$  and  $PN$  are shown as functions of time, rather than as sampled values.) Note that correlators 20 do not all start correlating at the same time. In this example, the first correlator 20 starts correlating at time  $t=0$ ; the second correlator 20 starts correlating at time  $t=T_c/2$ ; the third correlator 20 starts correlating at time  $t=T_c$ ; and the fourth correlator 20 starts correlating at time  $t=3T_c/2$ . Note also that, in this example at least, each correlator 20 receives the PN sequence with a delay corresponding to the time at which that correlator 20 starts its calculation. After M chip durations  $T_c(K=M)$ ,  $S_k=S_M$  is the first dwell correlation value. After N chip durations  $T_c(K=M)$ ,  $S_k=S_M$  is the second dwell correlation value.

Similarly, clock 32 is not part of searcher 10, but is the system clock of the mobile station of which searcher 10 is a high level component. Clock 32 drives PN sequence generator 12 under the control of hold unit 26, as described below.

PN sequence generator 12 produces a new value  $PN_k$  every chip duration  $T_c$ . Each new term in the right hand side of equation (4) also is computed by each correlator 20 once every  $T_c$ . In any particular  $T_c$  interval, all correlators 20

5

receive from A/D converters 72 one of two different values  $RX_k$  but each correlator 20 receives from PN sequence generator 12, via delay line 14 and multiplexer 18, a different value  $PN_k$ , depending on the value of an index stored in index register 22 associated with that correlator 20.

Conceptually, once every  $T_c$  interval, each correlator 20 performs the multiplication  $RX_k PN_k^*$  and adds the complex product thus obtained to a correlation value stored in one of the complex registers in memory 24 associated with that correlator 20. Because the possible values of the  $PN_k$  samples are either +1 or -1, there is no need to actually perform multiplications. Instead, only additions or subtractions of the in-phase and quadrature components of  $RX_k$  are actually performed. This allows a significant reduction in the complexity and electrical current consumption of searcher 10.

For example, let  $A = \text{Re}(RX_k) + j\text{Im}(RX_k)$  and let  $B = \text{Re}(RX_k) - j\text{Im}(RX_k)$ . If  $\text{Re}(PN_k) = 1$  and  $\text{Im}(PN_k) = 1$ , then  $\text{Re}(RX_k PN_k^*) = A$  and  $\text{Im}(RX_k PN_k^*) = -B$ . If  $\text{Re}(PN_k) = 1$  and  $\text{Im}(PN_k) = -1$ , then  $\text{Re}(RX_k PN_k^*) = B$  and  $\text{Im}(RX_k PN_k^*) = -A$ . If  $\text{Re}(PN_k) = -1$  and  $\text{Im}(PN_k) = 1$ , then  $\text{Re}(RX_k PN_k^*) = -B$  and  $\text{Im}(RX_k PN_k^*) = -A$ . If  $\text{Re}(PN_k) = -1$  and  $\text{Im}(PN_k) = -1$ , then  $\text{Re}(RX_k PN_k^*) = A$  and  $\text{Im}(RX_k PN_k^*) = B$ . Instead of transferring  $RX_k$  directly from receiver 30 to correlators 20,  $RX_k$  is sent to an arithmetic unit (not shown) that computes A and B and sends A and B to the appropriate correlators 20. Each correlator 20 then adds  $\pm A$  or  $\pm B$  to the real part and the imaginary part of the correlation value, depending on the signs of the values of  $\text{Re}(PN_k)$  and  $\text{Im}(PN_k)$  concurrently provided by multiplexer 18 to that correlator 20.

Another method of avoiding actual multiplications exploits the fact that only the absolute values of the correlation values S are actually needed, to further reduce the number of calculations and achieve a further reduction in electrical current consumption by searcher 10. If the complex PN sequence of every correlator 20 is rotated 45°, then either the real part or the imaginary part of every  $PN_k$  sample is equal to zero. Each correlator 20 then adds either  $\pm \text{Re}(RX_k)$  or  $\pm \text{Im}(RX_k)$  to the real part or the imaginary part of S, depending on the sign of the non-zero component of  $PN_k$ , without the intervention of the arithmetic unit. The rotation as described implicitly divides the complex PN sequence by the square root of 2. If only the relative values of S are required, then the system software uses these values of S as produced by correlators 20. If the absolute values of S are needed, then the system software normalizes the values of S that it obtains from searcher 10 by multiplying those values by the square root of 2.

Each delay unit 16 receives the PN sequence, either directly from PN sequence generator 12 in the case of the first (leftmost) delay unit 16, or from the immediately preceding delay unit 16. Each delay unit passes the PN sequence, with a fixed delay D, to multiplexer 18 and (except for the last (rightmost) delay unit 16) to the next delay unit 16. PN sequence generator 12 also passes the PN sequence directly to multiplexer 18. Thus, if there are  $N_D$  delay units 16 in delay line 14, multiplexer 18 receives  $N_D + 1$  copies of the PN sequence, with mutual relative delays D. The size of D, and the sampling rate at which  $RX_k$  samples are provided to correlators 20, are selected to give searcher 10 the required time resolution. In the example of equations (5)-(8), in which the sampling rate of  $RX_k$  is  $(T_c/2)^{-1}$ , the time resolution of searcher 10 is  $T_c/2$ .

Searcher 10 functions under the overall control of the system software to search for the delays, in all the relevant windows, that give correlation values that are significantly meaningful (i.e., above background noise) to be useful in

6

identifying the strong neighboring base stations and in demodulating the signals received from these base stations. For each window, the search process is initialized by setting the delay of PN sequence generator 12 to the first (earliest) delay in the window, by setting the indices stored in index registers 22 to values corresponding to the first L delays in the window (L being the number of correlators 20), and by zeroing the complex registers of memories 24. Subsequently, hold unit 26 delays PN sequence generator 12 further, as described below. In all cases, hold unit 26 delays PN sequence generator 12 by blocking timing signals from clock 32.

Whenever a correlator 20 finishes a correlation over M chip intervals, next location unit 28 decides whether that correlator 20 should continue correlating at its current delay or should move to the next delay. FIG. 2 is a flow chart of this decision. If  $K = M$  (block 40), correlator 20 has finished the first dwell correlation, so the absolute value of  $S_K = S_M$  is compared to the first dwell threshold (block 42). If  $|S_M|$  is less than or equal to the first dwell threshold, the correlation at the current delay has failed, so correlator 20 is moved to the next delay that needs to be tested (block 48). If  $|S_M|$  exceeds the first dwell threshold, then correlator 20 stays at the current delay (block 46) and continues the summation of equation (4) until N terms  $RX_k PN_k^*$  have been summed. If  $K > M$  (block 40), then, in the general case of  $N > 2M$ , either correlator 20 is in the middle of computing the second dwell correlation value  $S_N (K < N)$  or correlator 20 has finished computing the second dwell correlation value ( $K = N$ ) (block 44). If correlator 20 is in the middle of computing SN, then correlator 20 remains at the current delay (block 50). Otherwise, correlator 20 is moved to the next delay that needs to be tested.

In the special case of  $N = 2M$ ,  $K > M$  implies  $K = N$ , so the "no" branch of block 40 leads directly to block 48.

Most preferably, the exact absolute value of  $S_M$  is not compared to the threshold. Instead the following piecewise linear approximation of  $|S_M|$ , which is based on a linear approximation of  $\sqrt{1+x^2}$ , and which is easier to implement in hardware than an exact numerical calculation of the absolute value of  $S_M$ , is used for the absolute value of  $S_M$ .

$$|S_M| \approx \max(|\text{Re}(S_M)|, |\text{Im}(S_M)|) + \min(|\text{Re}(S_M)|, |\text{Im}(S_M)|) / 4 \quad (9)$$

This approximation is sufficiently accurate for first dwell thresholding, and allows the implementation of the first dwell threshold decision in a hardware unit that is smaller, and consumes less electrical current, than would otherwise be necessary. By contrast, the exact absolute value of  $S_M$  is computed, for trial delays that pass the first dwell threshold, in software, so that the various  $|S_M|$ 's can be compared to determine the delays with the largest  $|S_M|$ 's. The fact that only a small number of trial delays pass the first dwell threshold keeps the associated computational load on the system software relatively low, with no sacrifice in accuracy.

Recall that each memory 24 includes several complex registers for storing  $S_K$ . The register depth, i.e., the number R of complex registers, depends on how often (multiple of  $MT_c$ ) an interrupt is generated to allow the reading of the most recently calculated value of S and the reading of the index value in the associated integer register. For example, if the interrupt is generated every  $2MT_c$ , then R should be at least 2, and in general if the interrupt is generated every  $yMT_c$  (y being an integer) then R should be at least as great as y. If  $y < R$ , then the R complex registers are activated cyclically, giving the system software more time to respond to interrupts. R and y are implementation-dependent parameters. There are several considerations in the selection of the

optimum values of  $y$  and  $R$ . Values of  $y$  and  $R$  that are too small put too much of a burden on system software. Large values of  $y$  and  $R$  require a correspondingly long delay line and a larger chip area devoted to memories 24. The preferred value of both  $R$  and  $y$  is 2. Most preferably, to minimize the burden on system software, an interrupt is issued to system software only when all correlators 20 have filled their respective memories 24.

Next location unit 28 also includes a next location register. At the start of correlation in a given window, the value in the next location register is set to the index corresponding to the first delay after the initial  $L$  delays. Subsequently, whenever block 48 is reached for a given correlator 20, the value stored in the next location register is:

(a) copied to index register 22 of that correlator 20 and then

(b) changed to the index corresponding to the delay immediately following the delay to which that correlator 20 has now been set.

Every  $yM$  chip intervals, while the interrupt service routine reads the output of searcher 10, the system software determines the delay of the locally generated PN sequence that is to be used now by each correlator 20, and signals hold unit 26 to pause PN sequence generator 12 until the timing of the generation of the PN sequence by PN sequence generator 12 matches the earliest delay of the forthcoming  $M$  chip intervals. At the same time, multiplexer 18 shifts the input of the PN sequence to each correlator 20 correspondingly, to preserve the continuity of input to each correlator 20. This allows the use of a delay line 14 that is much shorter than the window. Specifically, the minimum value of  $N_D$ , the number of delay units 16 in delay line 14, is

$$\frac{L}{2} \frac{N}{M} + \Delta$$

where  $\Delta$  is an implementation dependent parameter:  $\Delta = Ly/2$ , where  $y$  is the interrupt interval factor defined above.

Preferably, the components illustrated in FIG. 1 all are implemented in hardware. The details of such a hardware implementation will be obvious to those skilled in the art.

The following is an example of the functioning of searcher 10, with  $L=8$  correlators 20 and with  $D=T_c/2$ ,  $M=512$ ,  $N=3M=1536$  and  $y=2$ . In this example, the value of the indices in index registers 22 and in the next location register of next location unit 28 are given as (possibly fractional) multiples of  $T_c$ . In practice, because index registers 22 are integer registers, the values actually stored in index registers 22 are appropriate integral multipliers of  $D$ . Similarly, the delays are expressed as multiples of  $T_c$  relative to the center of the window. A correlator 20 is said to "fail the first dwell threshold" if that correlator 20 produces a first dwell correlation value  $S_M$  less than or equal in absolute value to the first dwell threshold, and to "pass the first dwell threshold" if that correlator 20 produces a first dwell correlation value  $S_M$  having an absolute value greater than the first dwell threshold. All correlators 20 have two complex registers in memories 24 for accumulating correlation values.

Following the IS-95 standard, the first correlation is performed at a delay of  $-30$ .

TABLE 1

Status at Time = 0				
correlator no.	status	new index value	corresponding delay	new value in next location register
1		0	-30	
2		1/2	-29 1/2	
3		1	-29	
4		1 1/2	-28 1/2	
5		2	-28	
6		2 1/2	-27 1/2	
7		3	-27	
8		3 1/2	-26 1/2	4

TABLE 2

Status at time = 512T <sub>c</sub>				
correlator no.	status	new index value	corresponding delay	new value in next location register
1	fail threshold	4	-26	4 1/2
2	fail threshold	4 1/2	-25 1/2	5
3	fail threshold	5	-25	5 1/2
4	fail threshold	5 1/2	-24 1/2	6
5	fail threshold	6	-24	6 1/2
6	fail threshold	6 1/2	-23 1/2	7
7	fail threshold	7	-23	7 1/2
8	fail threshold	7 1/2	-22 1/2	8

Note: All the correlators have failed the first dwell threshold. Therefore, all the index registers are incremented by 4.

TABLE 3

Status at time = 1024T <sub>c</sub>				
correlator no.	status	new index value	corresponding delay	new value in next location register
1	fail threshold	8	-22	8 1/2
2	fail threshold	8 1/2	-21 1/2	9
3	pass threshold	5	-25	9
4	fail threshold	9	-21	9 1/2
5	fail threshold	9 1/2	-20 1/2	10
6	pass threshold	6 1/2	-23 1/2	10
7	fail threshold	10	-20	10 1/2
8	fail threshold	10 1/2	-19 1/2	11

Note: Correlators 3 and 6 have passed the first dwell threshold. Therefore, these two correlators remain at their old delays, to continue correlating for the second dwell time. The other correlators, having failed the first dwell threshold, are set to the next delays.

Now, an interrupt is generated. Hold unit 26 performs a hold of  $5T_c$ , which is delay of the earliest correlator (correlator 3) relative to the start of the window, and 5 is subtracted from all the index values and from the value in the next location register.

TABLE 4

Status at time = 1536T <sub>c</sub>				
correlator no.	status	new index value	corresponding delay	new value in next location register
1	fail threshold	6	-19	6 1/2
2	fail threshold	6 1/2	-18 1/2	7
3	continue (2 <sup>nd</sup> )	0	-25	7